Atty. Dkt. 2380-778
P18390US

U.S. PATENT APPLICATION

Inventor(s): Masayuki ARIYOSHI
Invention: TURBO CODE DECODER WITH PARITY INFORMATION UPDATE

NIXON & VANDERHYE P.C.
ATTORNEYS AT LAW
1100 NORTH GLEBE ROAD, 8TH FLOOR
ARLINGTON, VIRGINIA 22201-4714
(703) 816-4000
Facsimile (703) 816-4100

SPECIFICATION

TURBO CODE DECODER WITH PARITY INFORMATION UPDATE

BACKGROUND

[0001] FIELD OF THE INVENTION

[0002] The present invention pertains to information theory, and particularly to a class of binary parallel concatenated recursive systematic convolutional codes known as turbo codes, as well as apparatus and method using turbo codes.

[0003] RELATED ART AND OTHER CONSIDERATIONS

[0004] Data transmitted over a channel data may be corrupted from its original form. One example is data transmitted over a wireless, e.g., radio frequency, interface between a transmitter and receiver (one of which may be a mobile or cellular telephone, for example). Various encoding techniques have been employed so that errors in a string of received data bits may be located and, if possible, corrected. Such error correction techniques typically employ utilization of error correction code(s) which operate upon the original data in order to generate additional information which is transmitted along with the original data. Inclusion of the additional information with the received data facilitates the location and possible correction of error bits by the receiver, so that the receiver may ultimately obtain the original data as transmitted by the transmitter.

[0005] Many types of codes have been utilized in the error correction branch of information theory. Turbo codes, introduced in 1993, are considered to have high error correction capability and good performance. *See*, e.g., C. Berrout, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error Correcting Coding and Decoding: Turbo Codes", Proc. ICC93, pp. 1064 – 1070, May 1993, and C. Berrout and A. Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo Codes", *IEEE Trans. On Communications*, Vol. 44, No. 10, pp. 1261-1271, Oct. 1996.

[0006] The basic structure of turbo codes is the parallel concatenation of two or more component codes applied to different interleaved versions of the same information sequence. Recursive systematic convolutional (RSC) codes are used in general as the component codes. The constraint length K for the RSC codes could be shorter than that of conventional convolutional codes to achieve the equivalent or better performance.

[0007] Fig. 10 illustrates the structure of an example turbo code encoder. The example encoder 1000 of Fig. 10 comprises two RSC encoders 1002_1 and 1002_2 concatenated in parallel. In view of there being two encoders (e.g., M = 2), the code rate of the illustrative turbo code encoder has a code rate R = 1/3, since (as an approximation) R = 1/(M+1)). The input to the first RSC encoder 1002_1 is the original information sequence d. The original information sequence is also applied to an interleaver 1004 to produce an interleaved version d' of the original information sequence d. The interleaved version d' of the information sequence is the input to the second RSC encoder 1002_2. Since the RSC is the systematic encoder, the outputs from the turbo encoder are sequences of systematic u (=d) and redundant parts $x_{(1)}$ (output from the first RSC encoder 1002_1) and $x_{(2)}$ (output from the second encoder 1002_2). In other words, the output of the encoder 1000 is in the form u_1, $x_{1(1)}$, $x_{1(2)}$, u_2, $x_{2(1)}$, $x_{2(2)}$, where u_k is the k^{th} systematic bit (i.e., data bit), $x_{k(1)}$ is the parity output from the first RSC encoder 1002_1 associated with the k^{th} systematic bit u_k; and $x_{k(2)}$ is the parity output from the second RSC encoder 1002_2 associated with the k^{th} systematic bit u_k.

[0008] The decoding procedure for the turbo codes is known as iterative decoding. An example conventional turbo decoder, shown as decoder 1100 in Fig. 11, is comprised of two component decoders 1102_1 and 1102_2 (corresponding to the RSC encoders 1002_1 and 1002_2, respectively); an interleaver 1104 (which is the same type interleaver as interleaver 1004 used in the corresponding turbo encoder); and, a de-interleaver 1106 (which also corresponds to the interleaver 1004). The two component decoders 1102_1 and 1102_2 are soft-input and soft-output (SISO) decoders. The outputs of the two component decoders 1102_1 and 1102_2 are likelihood information concerning the coded data sequence.

[0009] The inputs to the turbo decoder 1100 are the channel measurements made at the detector for the systematic part $y_{(0)}$, the redundant parts $y_{(1)}$ and $y_{(2)}$. These inputs correspond to u, $x_{(1)}$, and $x_{(2)}$, respectively. The component decoder computes, for the

k^{th} decoded bit d_k, the probability that this bit was 1 or 0, after the received symbol sequence $\bar{y} = \{y_{(0)}, y_{(1)}, y_{(2)}\}$ is given.

[00010] Computing this probability is equivalent to finding the a posteriori log likelihood ratio (LLR) $L(\hat{d}) = L(d_k \mid \bar{y})$, as shown by Expression (1).

$$L(d_k \mid \bar{y}) = \ln\left(\frac{P(d_k = 1 \mid \bar{y})}{P(d_k = 0 \mid \bar{y})}\right) \qquad \text{Expression (1)}$$

In Expression 1, $P(d_k=1)$ is the probability that the bit $d_k=1$, and $P(d_k=0)$ is the probability that the bit $d_k=0$. The input LLR for the data bit d to the component decoder, designated $L_{in}(\hat{d})$, is expressed by Expression (2).

$$L_{in}(\hat{d}) = y_{(0)} + L(d) \qquad \text{Expression (2)}$$

In Expression 2, $L(d)$ is the a priori LLR of the d. For a systematic code, it has been shown (by C. Berrout, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error Correcting Coding and Decoding: Turbo Codes", Proc. ICC93, pp. 1064 – 1070, May 1993) that the LLR equals to

$$\begin{aligned} L(\hat{d}) &= L_{in}(\hat{d}) + Le(\hat{d}) \\ &= y_{(0)} + L(d) + Le(\hat{d}) \end{aligned} \qquad \textit{Expression (3)}$$

[00011] In Expression (3) $Le(\hat{d})$, called the extrinsic LLR, represents extra knowledge that is gleaned from the decoding process. In the iterative decoding procedure, the extrinsic LLR calculated from the previous component decoder will be set as a priori LLR for the next component decoder. At the very first component decoding, $L(d)$ is set as 0. Since the component decoders 1102 are connected each other together with the interleaver 1104/de-interleaver 1106 as seen in Fig. 11, these operations are repeated. With more decoding iterations, the accuracy of the likelihood of the systematic bits $L(\hat{d})$ becomes better. After a number of iterations, finally the sign of $L(\hat{d})$ will be the decoder decision for the result, and the amplitude of $L(\hat{d})$ denotes the reliability of that decision. Thus, using turbo decoding, the error correcting performance is improved through the iterative procedure.

[00012] Further discussions of principles involved in turbo coding and decoding are provided in B. Sklar, "A Primer on Turbo Code Concepts," *IEEE Communications Magazine*, pp. 94 – 102, Dec. 1997; A. Ushirokawa, T. Okamura, N. Kamiya, and B. Vucetic, "Principles of Turbo Codes and Their Applications to Mobile communications," *IEICE Trans. On Fundamentals,* vol. E81-A, No. 7, July 1998; J.P. Woodard and L. Hanzo, "Comparative Study of Turbo Decoding Techniques: An Overview," *IEEE Trans. On Vehicular Technology*, vol. 49, no. 6, pp. 2208-2233, Nov 2000.

[00013] Characteristics and performance of turbo codes are shown in P. Jung, "Comparison of Turbo-Code Decoders Applied to Short Frame Transmission Systems," *IEEE Journal on Selected Areas on Communications*, vol. 14, no. 3, pp. 530-537, April 1996; P. Jung and M. Baβhen, "Results on Turbo-Codes for Speech Transmission in a Joint Detection CDMA Mobile Radio System with Coherent Receiver Antenna Diversity," *IEE Trans. on Vehicular Technology*, vol. 46, no. 4, Nov. 1997; H. Koorapaty, Y.E. Wang, and K. Balachandran, "Performance of Turbo Codes with Short Frame Sizes", Proc. of 1997; and P. Frenger, "Turbo Decoding for Wireless Systems with Imperfect Channel Estimates," *IEEE Trans. on Communications*, vol. 48, no. 9, September 2000. A turbo code decoding analysis is provided in D. Divsalar, S. Dolinar and F. Pollara, "Iteractive Turbo Decoder Analysis Based on Density Evolution," *IEEE Journal on Selected Areas in Communication*, vol. 19, no. 5, May 2001. All the foregoing are incorporated by reference in their entirety.

[00014] In any type of decoder (not just turbo decoders), the accuracy of the decoder is heavily dependent upon the reliability of the input for the decoder. What is sought, therefore, and an object of the present invention, is a technique for improving the accuracy of a turbo decoder by improving reliability of its inputs.

BRIEF SUMMARY

[00015] In a turbo code decoder, likelihood information for redundant parts of a received symbol sequence is used to update the redundant parts of the received symbol sequence for use by the decoder.

[00016] In one example embodiment, the redundant likelihood update unit comprises a soft output recursive systematic convolutional (RSC) encoder which uses the likelihood information for a systematic part of a received symbol sequence to generate likelihood information for a first redundant part of the received symbol sequence and to generate likelihood information for a second redundant part of the received symbol sequence. The soft output recursive systematic convolutional (RSC) encoder handles *a posteriori* log likelihood ratio (LLR) values.

[00017] In another example embodiment, the turbo code decoder comprises a soft output decoder which provides both likelihood information for a systematic part of the received symbol sequence and likelihood information for redundant parts of the received symbol sequence (i.e., likelihood values for a first redundant part of the received symbol sequence and likelihood values for a second redundant part of the received symbol sequence).

[00018] In the example embodiments, the redundant likelihood update unit comprises a first compare and update unit and a second compare and update unit. The respective compare and update units compare the likelihood information for their respective redundant part of the received symbol sequence with the original respective redundant part of the received symbol sequence, and obtain updated likelihood information for their respective redundant parts of the received symbol sequence.

[00019] In updating a redundant part of the received symbol sequence the redundant likelihood update unit first compares, on a bit by bit basis, a sign of (1) the likelihood information for a redundant part of the received symbol sequence with a sign of (2) a corresponding original redundant part of the received symbol sequence. If the signs for (1) and (2) are the same, an amplitude of the corresponding updated redundant part for the k^{th} symbol of the received symbol sequence becomes an amplitude of a larger of (1) and (2). On the other hand, if the signs for (1) and (2) are different, a sum of (1) and (2) is used as the amplitude for the corresponding updated redundant part for the k^{th} symbol of the received symbol sequence.

[00020] The redundant likelihood update units further comprise a timing and control unit which controls when the redundant information should be updated in the iterative decoding process, and which supervises, e.g., the correlation between the re-encoded

sequence of the redundant LLR and the source sequence of the systematic LLR. For example, the timing and control unit can determine timing for operation of the redundant likelihood update unit based on a signal-to-noise ratio (SNR) of received signals, or based on interim decoding error counts, or both.

[00021] Another aspect of the invention is a wireless radio frequency receiver which has an antenna and a base band signal processor. The base band processor has a demodulator (which provides a demodulated base band signal); an optional de-interleaver (which de-interleaves the demodulated base band signal); an optional rate de-matcher (which adjust the number of demodulated bits being matched with that for the turbo decoder); and a turbo decoder. The turbo decoder comprises a redundant likelihood update unit which uses likelihood information for redundant parts of the received symbol sequence to update the redundant parts of the received symbol sequence for use by the decoder.

[00022] One advantage which attends a turbo code decoder with its redundant likelihood update unit (RLU) unit is improvement in bit error rate (BER) or block error rate (BLER) performance. Another advantage is that the achievable BER/BLER can be improved (providing, e.g., reduce decoding computations (iterations) and decoding processing time (delay)).

BRIEF DESCRIPTION OF THE DRAWINGS

[00023] The foregoing and other objects, features, and advantages of the invention will be apparent from the following more particular description of preferred embodiments as illustrated in the accompanying drawings in which reference characters refer to the same parts throughout the various views. The drawings are not necessarily to scale, emphasis instead being placed upon illustrating the principles of the invention.

[00024] Fig. 1A and Fig. 1B are schematic views of different example embodiments of turbo code decoders.

[00025] Fig. 2A and Fig. 2B are schematic views of different embodiments of example redundant likelihood update units for use in the turbo code decoders of Fig. 1A and Fig. 1B, respectively.

[00026] Fig. 3 is a schematic view of an example soft output RSC encoder for use in the example redundant likelihood update unit of Fig. 2A.

[00027] Fig. 4 is a flowchart showing example basic steps performed by a compare and update function of an example redundant likelihood update unit.

[00028] Fig. 5 is a schematic view of a first example embodiment timing and control unit for use in an example redundant likelihood update unit.

[00029] Fig. 6 is a schematic view of a second example embodiment timing and control unit for use in an example redundant likelihood update unit.

[00030] Fig. 7 is a graph showing error rate performance comparisons for the turbo code decoder of Fig. 1A.

[00031] Fig. 8 is a graph showing error rate performance as a function of decoding iterations for the turbo code decoder of Fig. 1A.

[00032] Fig. 9 is a schematic view of a wireless receiver that features a turbo code decoder such as the turbo decoder of Fig. 1A or the turbo decoder of Fig. 1B.

[00033] Fig. 10 is a schematic view of a conventional turbo code encoder.

[00034] Fig. 11 is a schematic view of a conventional turbo code decoder.

DETAILED DESCRIPTION OF THE DRAWINGS

[00035] In the following description, for purposes of explanation and not limitation, specific details are set forth such as particular architectures, interfaces, techniques, etc. in order to provide a thorough understanding of the present invention. However, it will be apparent to those skilled in the art that the present invention may be practiced in other embodiments that depart from these specific details. In other instances, detailed descriptions of well-known devices, circuits, and methods are omitted so as not to obscure the description of the present invention with unnecessary detail.

[00036] The general turbo decoding algorithm with its iterative procedure improves the accuracy of likelihood information for the systematic part which is input to the component decoders. Yet in the conventional turbo decoders, as described above, only the likelihood information for the systematic part is updated through the iterations. But there are other inputs to the component decoders – the redundant parts. Now described by non-limiting, representative examples embodiments are turbo decoders having reliability enhanced by utilizing a redundant likelihood technique implemented, e.g., by a redundant likelihood unit (RLU). In these examples, the inputs to the turbo decoder are not only the systematic parts information, but also the redundant parts information which is updated effectively, so that their accuracy and the decoding performance is improved.

[00037] Fig. 1A is a schematic view of an example embodiment turbo code decoder 100 which utilizes a redundant likelihood update technique. The turbo code decoder 100 includes a redundant likelihood update unit 101 as well as two component decoders 102_1 and 102_2; an interleaver 104; and, a de-interleaver 106. The two component decoders 102_1 and 102_2 are soft-input and soft-output (SISO) decoders. The component decoders 102_1 and 102_2, interleaver 104, and de-interleaver 106 are configured or chosen to correspond to compatible elements in a turbo encoder which generates a coded symbol sequence which is to be decoded by turbo code decoder 100.

[00038] From the foregoing it will be appreciated that turbo code decoder 100 is an example of turbo decoder for a code with code rate of 1/3. However, it should be understood that the present invention is not confined or limited to any particular code rate, and that by modification (such as including a greater number of component decoders, etc.) the turbo code decoder 100 can be configured for handling codes of a different code rate. Moreover, the turbo decoder 100 can be used with turbo codes with R>1/(m+1), e.g. for turbo codes with puncturing.

[00039] The turbo code decoder 100 of Fig. 1A is fed a received symbol sequence $\bar{y} = \{y_{(0)}, y_{(1)}, y_{(2)}\}$, and thereafter operates in an essentially iterative manner. The systematic part $y_{(0)}$ of the received symbol sequence is applied both to a first input terminal of component decoder 102_1 and to a first input terminal of interleaver 104. The redundant parts $y_{(1)}, y_{(2)}$ of the received symbol sequence are applied to respective input terminals of RLU unit 101. Corresponding output terminals of RLU unit 101

apply updated redundant parts $y_{u(1)}$, $y_{u(2)}$ to input terminals of component decoder 102_1 and component decoder 102_2, respectively.

[00040] The operation of turbo code decoder 100 occurs in an iterative manner. A full iteration comprises two half-iterations. In a first half-iteration the component decoder 102_1 performs its functionality, while in a second half-iteration the component decoder 102_2 performs its functionality. For a given received symbol sequence $\bar{y} = \{y_{(0)}, y_{(1)}, y_{(2)}\}$, there may be several full iterations performed by turbo code decoder 100. After one or more iterations have been performed, operation of the RLU unit 101 is invoked between further iterations until such time as RLU unit 101 is no longer needed. Prior to invocation of the RLU unit 101, the redundant parts $y_{(1)}$, $y_{(2)}$ of the received symbol sequence are essentially passed through the RLU unit 101 to be the updated redundant parts $y_{u(1)}$, $y_{u(2)}$ for application to input terminals of component decoder 102_1 and component decoder 102_2, respectively. When invoked, RLU unit 101 generates a likelihood value $L(y_{(1)})$ for redundant part $y_{(1)}$ and an updated redundant part $y_{u(1)}$, as well as a likelihood value $L(y_{(2)})$ for redundant part $y_{(2)}$ and an updated redundant part $y_{u(2)}$.

[00041] Describing further now the structure of turbo code decoder 100 of Fig. 1A, extrinsic LLR $Le_2(d)$ taken from an output terminal of de-interleaver 106 is applied to an input terminal of component decoder 102_1. During the first half-iteration, the component decoder 102_1 generates an LLR $L_1(d)$ at a first output terminal and an LLR $Le_1(d)$ at a second output terminal.

[00042] In addition to receiving the systematic part $y_{(0)}$ of the received symbol sequence at its first input terminal, the interleaver 104 receives at its second input terminal the likelihood extrinsic value $Le_1(d)$ generated by component decoder 102_1. The interleaver 104 uses the systematic part $y_{(0)}$ of the received symbol sequence to generate an interleaved systematic part $y'_{(0)}$ and an interleaved likelihood extrinsic value $Le_1'(d)$, both of which are applied to respective input terminals of component decoder 102_2.

[00043] In the second half-iteration the component decoder 102_2 generates values $L_2'(d)$ and $Le_2'(d)$ at respective output terminals, and applies both values to respective input terminals of de-interleaver 106. The de-interleaver 106 yields both a bit $L_2(d)$ and

a bit $Le_2(d)$ at its respective output terminals. As mentioned previously, the bit $Le_2(d)$ is applied to component decoder 102_1. The bit $L_2(d)$ is applied to hard bit decision unit 108, which produces the bit $\hat{d}$.

[00044] The decoding performed by turbo decoder 100 can, as an alternative mode of operation, be stopped after a first half of an interaction rather than waiting until completion of the second half of an interaction. In the half interaction stopping mode, the output $L_1(d)$ generated by component decoder 102_1 is applied to hard bit decision unit 108 for producing $\hat{d}$.

[00045] In the embodiment of Fig. 1A the redundant likelihood update (RLU) unit 101 serves as a front end to the remainder of the turbo decoder. As hereinafter explained, the RLU unit 101 provides likelihood information with improved reliability for redundant parts $y_{(1)}$ and $y_{(2)}$ of a received symbol sequence $\bar{y} = \{y_{(0)}, y_{(1)}, y_{(2)}\}$. In other words, with the functionality of RLU unit 101 being placed in front of the remaining elements of turbo decoder 100, the updated redundant likelihood information y_{u1} and y_{u2} will be used when appropriate in the turbo decoder instead of y_1 and y_2.

[00046] Fig. 2A shows an example redundant likelihood update (RLU) unit 101 of a type which is suitable for use in the turbo code decoder 100 of Fig. 1A. The example RLU unit 101 comprises a soft output recursive systematic convolutional (RSC) encoder 202 which uses the likelihood information for a systematic part of a received symbol sequence to generate likelihood information for a first redundant part of the received symbol sequence and to generate likelihood information for a second redundant part of the received symbol sequence. In particular, the soft output RSC encoder 202 of RLU unit 101 receives $L_1(d)$ which was generated by component decoder 102_1 during the first half-iteration, and uses $L_1(d)$ to generate the likelihood $L(y_{(1)})$. Similarly, the soft output RSC encoder 202 of RLU unit 101 receives $L_2(d)$ which was generated by component decoder 102_2 during the second half-iteration, and uses $L_2(d)$ to generate the likelihood $L(y_{(2)})$. As explained subsequently, the soft output RSC encoder 202 handles *a posteriori* log likelihood ratio (LLR) values.

[00047] In RLU unit 101 shown in Fig. 2A, soft output RSC encoder 202 with its input of systematic LLR re-generates likelihood information for the redundant parts (for the first redundant part and the second redundant part). The transfer function of soft output

RSC encoder 202 corresponds with that of original binary RSC encoder, but the SISO-RSC encoder treats LLR values instead of binary digits. The soft output RSC encoder 202 can thus be realized by using log likelihood algebra instead of binary Boolean algebra.

[00048] The RLU unit 101 of Fig. 2A also comprises a first compare and update unit 204_1 and a second compare and update unit 204_2. The first compare and update unit 204_1 compares the likelihood information $L(y_{(1)})$ for the first redundant part of the received symbol sequence with the original first redundant part $y_{(1)}$ of the received symbol sequence and obtains an updated likelihood value $y_{u(1)}$ for the first redundant part of the received symbol sequence. In similar manner, second compare and update unit 204_2 compares the likelihood information $L(y_{(2)})$ for the second redundant part of the received symbol sequence with the original second redundant part $y_{(2)}$ of the received symbol sequence and obtains an updated likelihood value $y_{u(2)}$ for the second redundant part of the received symbol sequence. Either one or both of compares and update units 204, and 204_2 may be in operation, but in either case control unit 210 acts as the master.

[00049] The operation of RLU unit 101, including the operation of soft output RSC encoder 202 and the compare and update units 204, is controlled and supervised by a timing and control unit 210. The timing and control unit 210 controls when the redundant information should be updated in the iterative decoding process, and supervises the correlation between the re-encoded sequence of the redundant LLR and the source sequence of the systematic LLR. That is, the timing and control unit 210 manages detection and updating of the error symbols, thereby enabling the redundant LLR value to improve the turbo decoding performance. Thus, it is important that the timing and control unit 210 operate the proposed update at an adequate timing. For example, an update at a very early stage of the decoding iterations could not work well, because the re-generated redundant LLR sequence is supposed to contain relatively more errors in lower SNR. On the contrary, if the redundant LLR sequence is re-generated at a considerably later stage, most of the errors might have been corrected through the decoding iterations. But this latter case is not expected to improve the error floor characteristics due to the strong correlation between the regenerated redundant and the source systematic sequence.

[00050] Fig. 3 shows an example soft output RSC encoder 202 for use in the example redundant likelihood update unit 101 of Fig. 2A. The example SISO-RSC encoder 202 of Fig. 3 is based on the following assumptions: the coding rate is R=1/3; RSC polynomials are 15, 13 in octal; and, the constraint length is K=4. In the soft output RSC encoder 202 of Fig. 3, the letter "D" denotes a shift register while the operator "+" denotes a log-likelihood addition. Thus, the example soft output RSC encoder 202 of Fig. 3 comprises four adders 302, 304, 306, and 308, and three shift registers 310, 312, and 314. Each of the four adders perform a log-likelihood addition operation.

[00051] The example soft output RSC encoder 202 of Fig. 3 may operated so that the soft output RSC encoder 202 receives $L_1(d)$ as $L_n(U)$, and outputs $L(y_{(1)})$ as $L(y_{(1)})$, and subsequently receives $L_2(d)$ as $L_n(U)$, and outputs $L(y_{(2)})$ as $L(y_{(2)})$. Alternatively, two separate and independent channels of the structure illustrated in Fig. 3 may be provided, one channel for the first redundant part and a second channel for the second redundant part.

[00052] In the soft output recursive systematic convolutional (RSC) encoder 202 of Fig. 3, a first shift register 310 receives an addition result from the first adder 302. A second adder 304 receives the likelihood information for a systematic part of a received symbol sequence as a first input to the second adder 304. A first shift register 310 receives an addition result from the first adder 302; a second shift register 312 receives shifted-out contents of the first shift register 310. The shifted-out contents of the second shift register 312 are also applied as a second input to the second adder 304. A third shift register 314 receives shifted-out contents of the second shift register 312. A third adder 306 receives the shifted-out contents of the second shift register 312 as a first input to the third adder 306 and receives shifted-out contents of the third shift register 314 as a second input to the third adder 306. An addition result of the third adder 306 is applied as a second input to the first adder 302. A fourth adder 308 receives an addition result from the second adder 304 as a first input to the fourth adder 308 and receives the shifted-out contents of the third shift register 314 as a second input to the fourth adder 308. An addition result of the fourth adder 308 is the likelihood value $L(y_{(i)})$ for one of the redundant parts of the received symbol sequence.

[00053] Thus, log-likelihood addition is performed by the adders of soft output RSC encoder 202. The log-likelihood addition is performed as an alternative to modulo-2

addition, modulo-2 addition being denoted by the operator $\oplus$. The log-likelihood addition is performed for two log-likelihood inputs "a" and "b" by the adders of soft output RSC encoder 202 is described by Expression (4).

$$L(a) \boxplus L(b) \equiv L(a \oplus b)$$
$$\approx (-1) \times sign[L(a)] \times sign[L(b)] \times min[\,|L(a)|, |L(b)|\,] \qquad (4)$$

[00054] With the approximations resulting from use of Expression (4), the amplitude of the operation result will be the minimum of the two inputs. If there are error symbols in the input sequence, those errors are spread into the coded sequence afterwards, due to the recursive encoding. Such spreading of errors is a risk when using the re-encoding method in the lower signal to noise ratio (SNR) conditions. However, the amplitude of the error symbols is statistically small and hence the amplitude of re-encoded LLR values are limited as minimum by the log-likelihood addition.

[00055] Fig. 1B shows another example embodiment turbo code decoder 100B which differs from the turbo code decoder 100 of Fig. 1A in that, e.g., the component decoder $102B_1$ and the component decoder $102B_2$ of turbo code decoder 100B have the additional capability of generating the likelihood information for the redundant parts of the received symbol sequence. In particular, the component decoder $102B_1$ generates likelihood information (shown as $Ly_{(1)}(d)$ in Fig. 1B) for a first redundant part of the received symbol sequence. Similarly, component decoder $102B_2$ generates likelihood information (shown as $Ly_{(2)}(d)$ in Fig. 1B) for a second redundant part of the received symbol sequence. Such being the case, the companion RLU unit 101B (see Fig. 2B) does not require a RSC encoder. Rather, the RLU unit 101B can have the likelihood information for the first redundant part (shown as $L(y_{(1)})$ in Fig. 2B) applied directly to the first compare and update unit 204_1, and the likelihood information for the second redundant part (shown as $L(y_{(2)})$ in Fig. 2B) applied directly to the second compare and update unit 204_2. In other respects, the structure and operation of the turbo decoder 100B of Fig. 1B and the RLU unit 101B of Fig. 2B are understood from the foregoing analogous description of the first embodiment.

[00056] Thus, for the turbo code decoder 100B of Fig. 1B, the component SISO decoders $102B_1$ and $102B_2$ provide the likelihood information for redundant part $L_{y1}(d)$ or $L_{y2}(d)$ as well as for systematic part $L_1(d)$ or $L_2(d)$. In this embodiment, component

SISO decoders $102B_1$ and $102B_2$ render the SISO-RSC encoder 202 unnecessary. Therefore, as shown in Fig. 2B, the RLU unit 110B does not include a SISO-RSC encoder. The redundant likelihood update (RLU) unit 110B serves as a front end to the remainder of the turbo decoder, and supplies updated likelihood information for redundant part y_{u1} and y_{u2} and thereby provides improved reliability.

[00057] Fig. 4 is a flowchart showing example basic steps performed by a compare and update function of the example redundant likelihood update unit of Fig. 2A or the example redundant likelihood update unit of Fig. 2B. It should be understood that the general steps shown in Fig. 4 are representative of those performed by either first compare and update unit 204_1 or the second compare and update unit 204_2.

[00058] In updating a redundant part of the received symbol sequence, as step 4-1 a comparison is performed, on a bit by bit basis, between a sign of (1) the likelihood information $L(y_{(i)})$ for a redundant part (as obtained from soft output RSC encoder 202), and a sign of (2) a corresponding original redundant part $y_{(i)}$ of the received symbol sequence. If the signs for both $L(y_{(i)})$ and $y_{(i)}$ are the same, then as step 4-2 the updated redundant part for the k^{th} symbol of the received symbol sequence, e.g., $y_{u(i)}$ is formed by using that same sign and by using, for the amplitude of the updated redundant part $y_{u(i)}$, the amplitude of the larger of $L(y_{(i)})$ and $y_{(i)}$. On the other hand, if the signs for $L(y_{(i)})$ and $y_{(i)}$ are different, at step 4-3 an optional check is made to ascertain whether an assumption has been made that the re-encoded sequence (e.g., $L(y_{(i)})$ as generated by soft output RSC encoder 202) is error free. If the check of step 4-3 reveals that the re-encoded sequence (e.g., $L(y_{(i)})$ as generated by soft output RSC encoder 202) is indeed assumed to be error free, then as step 4-4 the re-encoded sequence values (e.g., $L(y_{(i)})$ are utilized for the updated redundant part of the received symbol sequence, $y_{u(i)}$. If no such assumption has been made regarding the integrity of the re-encoded sequence $L(y_{(i)})$, or if optional step 4-3 is not performed, step 4-5 is executed. At step 4-5, a sum of $L(y_{(i)})$ and $y_{(i)}$ is utilized for the corresponding updated redundant part $y_{u(i)}$.

[00059] The compare and update units 204 thus first compare the redundant LLR sequence generated by re-encoding systematic LLR (by soft output RSC encoder 202) with the channel measurement for error detection. After the comparison, the compare and update units 204 update the original channel measurement values in response to the error detection results. Assuming that the LLR sequence input to the soft-RSC re-

encoder 202 is error free in signs, the re-encoded sequence can be treated as the redundant likelihood information whose signs are error free. In such case, the signs of re-encoded sequence of redundant likelihood are compared with those of the channel measurement sequence for error detection. The comparisons are made for each bit in the sequence. Then the likelihood information is updated either at step 4-2 or step 4-5 according to the error detection results. In the case those signs are the same (determined at step 4-1), at step 4-2 the updated sequence will take either of larger amplitude value, namely for the k^{th} symbol in the sequence, as indicated by Expression (5).

$$y_{ui}(k) = \begin{cases} y_i(k) & for |y_i(k)| > |L(y_i(k))| \\ L(y_i(k)) & otherwise \end{cases} \qquad \text{Expression} \quad (5)$$

Step 4-2 utilizes Expression (5) for two reasons: the matched signs should be recognized as probable, and the likelihood information should prevent their amplitude from getting too small by the effect of the SISO-RSC encoder 202.

[00060] If it is determined at step 4-1 that the signs are the different, either the channel redundant likelihood or the re-encoded likelihood is in error. If at step 4-3 it is guaranteed that the re-encoded sequence is error free, as step 4-4 the updated value takes the re-encoded likelihood. However, in reality there might be errors in the re-encoded sequence. Thus, if the updated value would take the re-encoded likelihood in such cases, the error symbols will become difficult to be corrected even though the decoding iteration goes because the re-encoded sequence is correlated with the source sequence which is the interim decoding result. That will cause the error floor characteristics. Therefore, the updated value should be neutral between the re-encoded likelihood and channel value as long as additional information about the reliability will not be obtained. Therefore, in one embodiment reflected in Fig. 4, as step 4-5 the updated sequence takes the sum of the LLR values of channel measurements and of SISO-RSC outputs assuming the reliability of them are equally treated, in accordance with Expression (6).

$$y_{ui}(k) = y_i(k) + L(y_i(k)) \,. \qquad \textit{Expression} \quad (6)$$

[00061] By using the rules of Fig. 4, the updated $y_{ui}(k)$ becomes more reliable than the original $y_i(k)$, whether k is an error or not. Consequently, the error correcting capability of the turbo code decoder is significantly improved.

[00062] Several solutions are possible for implementing the RLU timing control, e.g., for implementing timing and control unit 210. One simple solution is based on the signal-to-noise ratio (SNR) of received signals. Fig. 5 illustrates an embodiment of timing and control unit 210-5 utilizing such a solution, wherein timing and control unit 210 comprises (or consults) a pre-defined timing information table 500. When consulted via table handler logic 502, the table 500 supplies the optimal timings for the RLU update in accordance with the conditions of received SNR (the SNR conditions being used as an index to obtain the timing information for RLU unit 101). Once the information of SNR for received signals $y_{(0)}$, $y_{(1)}$, and $y_{(2)}$ has been obtained, the corresponding optimal timing information for the RLU update can be obtained by table handler 502 from table 500 in lookup fashion.

[00063] Another example embodiment for implementing a timing and control unit for RLU unit 101 is shown in Fig. 6. The timing and control unit 210-6 of Fig. 6 determines timing for operation of RLU unit 101 based on interim decoding error counts. In this embodiment, an estimate or count interim decoding errors in decoding iterations is applied to timing and control unit 210-6. The timing and control unit 210-6 comprises (or consults) a pre-defined timing information table 600. When consulted via table handler logic 602, the table 600 supplies the optimal timings for the RLU update in accordance with the condition (e.g., number or variation) of interim decoding errors. The decoder of this embodiment either employs or works in conjunction with means to estimate or count interim decoding errors in decoding iterations. According to the condition of interim decoding errors, the appropriate RLU timing information can be ascertained (e.g., from table 600). For example, a variation of error counts (decreasing or increasing) in time-series (decoding iterations) can be used as input.

[00064] Other configurations of timing and control unit 210 are also possible. For example, another embodiment of timing and control unit 210 can determine timing information based on both variation of error counts and received SNR measurement(s). Further, a method of determining timing for the RLU unit 101 using cross-entropy between outputs from two component decoders is also possible. Cross-entropy has

previously been proposed as the stopping criterion of decoding iterations (*see*, M. Moher and T.A. Gulliver, "Cross-Entropy and Iterative Decoding," *IEEE Trans. On Information Theory*, vol. 44, no. 7, Nov. 1998).

[00065] The RLU unit 101 and/or individual function blocks comprising RLU unit 101 may be implemented using individual hardware circuits, using software functioning in conjunction with a suitably programmed digital microprocessor or general purpose computer, using an application specific integrated circuit (ASIC), and/or using one or more digital signal processors (DSPs).

[00066] Since the turbo code decoder 100 with its RLU unit 101 is independent from the choice of the component decoder algorithms, there are no limitations for the selection of the component decoder, e.g., component decoder 102_1 and component decoder 102_2 can operate in accordance with any algorithm, e.g., SOVA. MAP, Log-MAP. Max-Log-MAP, etc.

[00067] One advantage which attends the turbo code decoders 100, 100B with their respective RLU units 101, 101B is improvements in bit error rate (BER) or block error rate (BLER) performance. Fig. 7 shows BER and BLER performance as a function of average SNR (E_b/N_0), where the conditions of the evaluation are assumed as listed in Table 1, and for an assumed Additive White Gaussian Noise (AWGN) channel model.

Table 1: Evaluation Parameters

Items	Conditions
Coding Rate	R=1/3
RSC Polynomials	15, 13 (octal), K=4 [13]
Interleaver	Prime Interleaver [13]
Block Size	656 bits (plus 3 tail bits)
Component Decoding Algorithm	BCJR (Log-MAP) [14]
Trellis Termination Strategy	Addition of 3 tail bits [13]

Regarding the specific parameter set for the proposed decoding algorithm such as redundant likelihood update timing, the values optimized from the prior evaluations are

used, i.e., RLU is carried out after the 1st decoding iteration for E_b/N_0=1dB and 2dB, and after the 7th iteration for 0dB. It is seen that the proposed decoding method achieves improvements on the error rate performance, particularly in the regions of BER less than 10^{-2}, and BLER less than 10^{-1}. The results show that the proposed method of updating redundant likelihood information works effectively.

[00068] As another advantage, the achievable BER/BLER can be improved by the turbo code decoders 100, 100B with their RLU units 101, 101B. In other words, the turbo code decoders with their RLU units can reduce decoding computations (iterations) and decoding processing time (delay). Fig. 8 shows the BER transition with increment of iteration numbers. In the case of average E_b/N_0=0dB, BER for the proposed decoding method is slightly better than that for the conventional method after the 8th iteration, because the redundant LLR update timing is the 7th iteration. In the case of average E_b/N_0=1dB, remarkable improvement can be seen on the error rate performance by the proposed decoding method with redundant likelihood update. Particularly, the BER/BLER curves for the proposed method drop quickly as number of iterations increases. This implies that the likelihood information of error bits potentially causing the error floor characteristics, is effectively corrected by the proposed method. In this condition, the proposed method improves the achievable BER/ BLER limit, and reaches to the possible BER/ BLER target at the earlier iterations than that of the conventional method. Since the channel condition of average Eb/N_0=2dB seems good enough, no significant differences are observed on the error rates of the proposed and the conventional decoding methods. Only a little improvement with the proposed method is seen at the 4th decoding iteration afterwards. From these results, it is found that there exist the regions where the noisy information on the redundant parts from the channel measurement might cause error floor characteristics. In that regions the proposed method updating the likelihood information for the redundant parts can improve error correcting performance of turbo codes.

[00069] The turbo code decoders herein described with their respective RLU units can be utilized in wide areas of applications that employ turbo codes. Nowadays, turbo codes have been in various practical use, such as channel coding scheme for cellular telecommunication systems, W-CDMA, cdmaOne, cdma2000, etc., wireless LAN,

video transmission systems with MPEG-4, and etc. The equipment for all these systems could utilize the turbo decoder of the present invention.

[00070] For example, Fig. 9 shows a wireless receiver 900 that features turbo code decoder 100-10. The wireless receiver 900 may be a receiver part of a network node such as a base station, or part of a mobile station such as a cellular phone, user equipment unit, or laptop (or other device) with mobile termination. The wireless receiver 900 has an antenna 902 and a base band signal processor/processing function 904. The base band processor 904 includes a demodulator 906 (e.g., rake receiver, which provides a demodulated base band signal); a de-interleaver 908 (optional, which de-interleaves the demodulated base band signal); rate de-matcher 909 (optional, which adjusts the number of demodulated bits being matched with that for the turbo decoder), and, turbo decoder 100-10. The decoded output from turbo code decoder 100-10 can optionally be input to a cyclic redundancy check (CRC) 910 for error detection.

[00071] As shown in Fig. 9, the turbo code decoder 100-10 comprises redundant likelihood update unit 101-10 which, in like manner as RLU unit 101 of Fig. 2A or the RLU unit 101B of Fig. 2B, uses the likelihood information for a systematic part of a received symbol sequence to generate likelihood information for redundant parts of the received symbol sequence, and uses the likelihood information for redundant parts of the received symbol sequence to update the redundant parts of the received symbol sequence for use by the decoder.

[00072] In a conventional turbo decoder, no likelihood information correlating to the redundant parts of the received symbol sequence is provided. By contrast, advantageously turbo code decoder 100 employs its RLU unit101, featuring soft output RSC encoder 202, which re-generates LLR values for y_i as $L(y_i)$, from its input $L_i(d)$, which input is known as the LLR valve for the systematic part. The SISO-RSC encoder 202 treats LLR values as soft-input and soft-output, but the functions of both SISO-RSC encoder and original binary RSC encoder correspond to each other. The LLR values for redundant parts generated by the SISO-RSC encoder 202 are compared with the sequence of original channel measurement for each symbol and updated.

[00073] The turbo decoder or decoding method described in example fashion above utilizes likelihood information for redundant parts of the received symbol sequence as

inputs to the decoder, thereby providing improved reliability especially for redundant parts. Specifically, the redundant parts' likelihood information is updated to improve reliability.

[00074] Thus, in a first aspect of the invention, there is provided a turbo code decoder or decoding method which employs a Redundant Likelihood Update (RLU) unit or function in front end to a conventional turbo decoder. In the RLU, an interim likelihood information of the systematic part output from a component SISO decoder is re-encoded, thereby reflecting their likelihood information. Then the re-generated likelihood information for the redundant parts is compared with the redundant information already input to the decoder, and the updated values are generated accordingly. The updated value of redundant likelihood in the RLU can be more accurate and help the iterative decoders to improve their performance of coding gain and achievable BER limit.

[00075] In another aspect of the invention, there is provided a turbo code decoder or decoding method which employs a modified type of the component SISO decoders providing likelihood information for both systematic and redundant parts. In the modified SISO decoders, the likelihood information for redundant part is also updated through the decoding iterations.

[00076] Additionally in yet another aspect of the invention, there is provided a turbo code decoder or decoding method which employs another type of RLU unit or function in front end and the modified component SISO decoders described above. In this case, RLU controls the timing when the updated likelihood information for the redundant part is input to the component SISO decoders.

[00077] While the invention has been described in connection with what is presently considered to be the most practical and preferred embodiment, it is to be understood that the invention is not to be limited to the disclosed embodiment, but on the contrary, is intended to cover various modifications and equivalent arrangements included within the spirit and scope of the appended claims.